今天学习的内容

1:复习近几个阶段的知识重点

2:项目

3:今天的目标

3.1:nodejs 中间件 --(重点&难点)

Express;中间件(static);

中间件(Middleware)是一个函数(session;cors;cookie)

作用:它可以访问请示对象(request object) 也可以用于访问响应对象(resonse object)和web应用请求-响应中对象.

中间件的功能包括

(1)执行任何代码

(2)修改请求和响应对象

(3)终结请求-响应循环

(4)调用下一个中间件

中间件分类

(1)应用级中间件(拦载器)

(2)路由级中间件 路由=请求路径+处理方法+处理函数

app.get("/user",(req,res)=>{});

(3)错误处理中间件

(4)内置中间件(express static)

(5)第三方中间件(session;cookie; cors)

中间件语法:

var app = express();

#req 请求对象

#res 响应对象

#next 调用下一个中间件

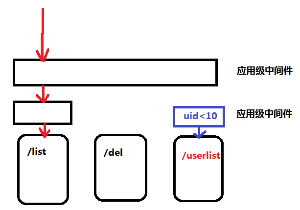
app.use(function(req,res,next){

console.log("中间任何代码");

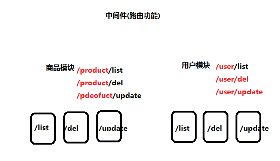
next();

});

应用级中间件



路由级中间件



//统一定义产品路由中间件

var router = express.Router();

router.get("/list",(req,res,next));

router.get("/del",(req,res,next));

router.get("/update",(req,res,next));

//统一定义用户路由中间件

var router1 = express.Router();

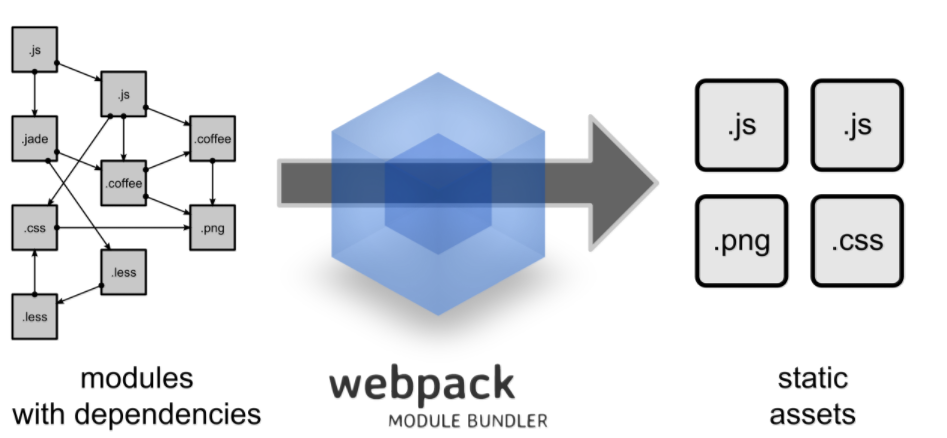
router1.get("/list",(req,res,next));

//加载应用

app.use("/product",router);

app.use("/user",router1);

3.2:webpack



webpack是一款加载器和打包工具:它能把各种资源(js/less/coffect/图片)等作为模块来使用处理并且打包.

安装和配置

1:创建测试项目 e:/webpack-test

2:进入项目目录中 e:

mkdir webpack-test

cd webpack-test

3:初始化项目 npm init

4:全局安装 npm install webpack -g

5:本地安装 npm install webpack --save-dev

示例:

1:创建js webpack-test/test.js

2:添加内容 hello

3:webpack test.js test.bundle.js

示例:

1:创建js webpack-test/word.js

2:添加

3:test.js 中添加文件 requre("./word");

4:webpack test.js test.bundle.js

示例

1:创建css 1.css

2:test.js 引入 1.css

3: webpack test.js test.bundle.js

错误:安装依装第三方模块(loader)

npm install css-loader style-loader --save-dev

并且修改过代码

示例:

(1)监控项止，内容实时修改

webpack test.js test.bundle.js --watch

(2)看打包过程

webpack test.js test.bundle.js --progress

(3)使用哪些模块

webpack test.js test.bundle.js --display-modules

示例:

(1)index.html 引入打包后文件

示例:

创建目录 e:/webpack-demo

dist #生成静态资源文件

node\_modules

src #源程序

script

styles

package.json 描述文件

webpack.config.js webpack配置文件

1:添加src/script/main.js

2:添加webpack.config.js

3:打包 webpack

示例:修改描述文件 package.json

添加入口

"webpack":" webpack --progress --colors "

再打包

npm run webpack